Orquestación de recursos para realizar cómputo de big data sobre arquitecturas distribuidas

María Murazzo[1], Nelson Rodríguez[1], Miguel Guevara[1], Martin Tello[4], Matías Alonso[5], Diego Medel[1], Jorge Mercado[2], Pablo Gómez[5], Fabiana Picolli[3]

[1] Departamento e Instituto de Informática - F.C.E.F. y N. - U.N.S.J.
[2] Departamento de Matemática, Facultad de Ingeniería - U.N.S.J.
[3] Departamento de Informática - F.C.F.M. y N, Universidad Nacional de San Luis
[4]Alumno Avanzado Licenciatura en Cs. de la Computación - F.C.E.F. y N. - U.N.S.J.
[5] Alumno Avanzado Licenciatura en Sist. de Información - F.C.E.F. y N. - U.N.S.J.

Complejo Islas Malvinas. Ignacio de la Roza y Meglioli. C.P. 5402. Rivadavia. San Juan, 0264 4234129

marite@unsj-cuim.edu.ar, nelson@iinfo.unsj.edu.ar, migueljoseguevaratencio@gmail.com, martinl.tello@gmail.com, matialonso95@gmail.com, vdiego.unsj@hotmail.com, jnmercado@unsj-cuim.edu.ar, mpiccoli@unsl.edu.ar

Resumen

Los avances tecnológicos, en particular IoT, han favorecido la generación de grandes cantidades de datos, los cuales necesitan ser almacenados y procesados de manera eficiente. Surge así el paradigma Big Data, donde el principal requerimiento no solo es la capacidad de cómputo, sino el manejo en un tiempo razonable de ingentes cantidades de datos. En este contexto, las aplicaciones para big data necesitan ser escalables, livianas, autocontenidas, distribuidas y replicadas con el objetivo de lograr la mejor performance frente a variaciones del volumen de datos.

Para lograr esto, esta línea de investigación propone ajustar la construcción de aplicaciones a la filosofía DevOps y una arquitectura basada en microservicios los cuales puedan ser implementados con contenedores. La replicación y distribución para lograr altos niveles de escalabilidad se plantea mediante la orquestación de contenedores sobre una arquitectura distribuida, la cual puede ser física o virtual. De esta manera será posible desarrollar aplicaciones que escalen de forma elástica en función de los requerimientos de la aplicación misma, independiente del software y hardware subyacente.

Palabras clave*: Big Data, Micrsoservices, Conteiners, DevOps, Distributed Systems.*

Contexto

El presente trabajo se encuadra dentro del área de I/D Innovación en Sistemas de Software y se enmarca dentro del proyecto de investigación: Orquestación de servicios para la Continuidad de Edge al Cloud, que ha sido aprobado y está en desarrollo para el período 2018-2019. Asimismo el grupo de investigación viene trabajando en proyectos relacionados con la computación móvil, distribuida y de alta performance desde hace más de 18 años. En esta oportunidad se han incorporado investigadores de otras universidades de la región lo cual impactará en todas las actividades planificadas. Las unidades ejecutoras para dicho proyecto son el Departamento e Instituto de Informática de la FCEFyN de la UNSJ.

Introducción

La integración de Cloud Computing (CC) [1] con Internet of Thing (IoT) [2] representa el próximo gran paso en las TI del futuro. Las aplicaciones que surjan de esta integración abrirán nuevas perspectivas de negocio y oportunidades de investigación. Gracias al paradigma CloudIoT [3], la vida cotidiana mejorará, por ejemplo, en las Smart city [4], permitirán servicios públicos más eficientes través de aplicaciones ubicuas que mejorarán la calidad de vida de los ciudadanos. El futuro

de este paradigma pasa por la convergencia hacia una plataforma de servicio común que supere los desafíos planteados por la heterogeneidad de los dispositivos y las tecnologías involucradas. Esto conlleva tener en cuenta la ubicuidad y la omnipresencia de los dispositivos, lo cual, requieren plataformas de computación escalables capaces de asegurar la continuidad hacia el cloud [5] .

La aceleración en la tasa de generación de datos a dando lugar al paradigma Big Data [6], el cual se caracteriza por tres aspectos: a) el volumen de los datos, b) los datos no se pueden almacenar de forma tradicional y c) los datos son generados, capturados y procesados con rapidez. Estas características hacen que los sistemas de cómputo, las metodologías de desarrollo y los framework de desarrollo convencionales sean inapropiados para lograr una adecuada gestión del big data [7].

Construir aplicaciones para big data tiene como principal problema la necesidad de administrar recursos en forma elástica, lo cual implica analizar y planificar la forma en la cual la aplicación debe escalar. Tradicionalmente, la construcción de aplicaciones ha seguido un paradigma de escalado vertical mediante el cual la solución a los problemas de cuellos de botellas se resuelve mediante el aprovisionamiento de mayor cantidad de recursos. Pero cuando se debe almacenar, procesar y analizar grandes cantidades de datos, las aplicaciones deben pensarse y construirse de forma distribuida; obligando a contar con mecanismos de escalabilidad horizontal, que permita distribuir la carga de trabajo dinámicamente y paralelizar las tareas [8] [9] .

La escalabilidad horizontal implica balanceo de carga, replicación de recursos, reinstanciación de recursos en tiempo real y optimización en su uso, de forma que el usuario no perciba degradación de performance [10]. Para lograr esto es necesario migrar la forma en la cual se piensan y se construyen las aplicaciones con el objetivo que se alineen con la misión y la visión organizacional.

Arquitectura de Software

Tradicionalmente, el desarrollo de software se ha ajustado en una arquitectura monolítica, donde las aplicaciones poseen una única base de código, la cual ofrece una variedad de servicios, que utilizan diferentes interfaces tales como páginas HTML encapsuladas a un mismo repositorio. La desventaja que presenta este enfoque es que, si se necesita realizar una modificación sobre una funcionalidad, impactará en las demás funcionalidades del sistema de modo tal que aumenta el costo de llevar a cabo dicha tarea. Otra desventaja es que, al tratarse de un esquema único, pueden generar un cuello de botella debido a que un error en la aplicación deja sin servicios todo el sistema.

Trabajar con grandes volúmenes de datos provenientes del IoT con las restricciones generadas por la necesidad de asegurar la continuidad al cloud, hacen inviable aplicar este modelo arquitectónico. Es por ello que resulta apropiado trabajar con arquitecturas basadas en servicios. En este caso, cada servicio son aplicaciones de negocio auto descriptivas y modulares que exponen la lógica organizacional como servicios a través de Internet por medio de una interfaz programable y donde el protocolo de Internet (IP) puede ser utilizado para encontrar e invocar esos servicios. Un servicio es el elemento clave en la integración de diferentes sistemas de información, ya que los sistemas de información pueden basarse en diferentes plataformas, lenguajes de programación y tecnologías. En particular, la arquitectura de microservicios [11] permite construir aplicaciones distribuidas como un conjunto de pequeños servicios desacoplados, desplegados de manera independiente y que trabajen coordinadamente.

Una de las características más importantes que provee los microservicios es la escalabilidad [12]. Debido a que los microservicios se desarrollan en forma independiente uno de otros, también escalan en forma independiente, esto es muy importante cuando se trabaja con grandes volúmenes de datos pues permite la implementación de carga compartida de trabajo.

Los microservicios dependen no solo de la tecnología que se está configurando, sino también de una organización que tenga la cultura, el conocimiento y las estructuras establecidas para que los equipos de desarrollo puedan adoptar este modelo. Los microservicios forman parte de un cambio más amplio en los departamentos de TI hacia una cultura DevOps [13], en la que los equipos de desarrollo y operaciones trabajan estrechamente para respaldar una aplicación a lo largo de su ciclo de vida y pasan por un ciclo de publicación rápido o incluso continuo en lugar de un ciclo tradicional largo.

Contenerización

El uso de microservicios implica la construcción de aplicaciones a partir de múltiples componentes autosuficientes. Estos componentes pueden ser implementados con contenedores [14]. Un contenedor, es un paquete de software que, internamente, contiene la aplicación que se quiere ejecutar y todas sus dependencias usando indirectamente el kernel del sistema operativo subyacente. Los contenedores permiten implementar aplicaciones distribuidas debido a dos características: portabilidad, pues los contenedores se pueden ejecutar de forma independiente al hardware y al software donde se crearon; y baja sobrecarga, pues son livianos e introducen menos overhead que las VM (Virtual Machine) [15].

La principal ventaja [16] de utilizar contenedores es que no depende de las instalaciones del sistema anfitrión, es decir es posible realizar las instalaciones necesarias para desplegar aplicaciones dentro de un contenedor. Otra ventaja es que permite la portabilidad en las aplicaciones, esto quiere decir que además de crear contenedores, es posible definir repositorios que eventualmente alojarán a los contenedores con el objeto que otros usuarios los descarguen y los consuman como servicio.

Quizás la características más relevante de los contenedores es que permiten desplegar aplicaciones de forma rápida y escalables, ya que permite replicar ambientes de desarrollo, prueba y producción en poco tiempo, pudiendo aumentar o disminuir la cantidad de contenedores a medida que sean necesitados (escalabilidad elástica) [17].

El uso conjunto de contenedores y microservicios [18] mejora las capacidades de escalado, ya que el microservicio es portable y reutilizable; mientras que los contenedores proporcionan recursos eficientes, principalmente el empaquetado de aplicaciones y sus dependencias en un contenedor virtual que puede ejecutarse en cualquier servidor.

Orquestación de Servicios

Se ha dicho que cuando se trabaja con datos masivos es necesario escalar horizontalmente las aplicaciones, esto se puede lograr mediante el uso de arquitecturas distribuidas con el objeto de repartir los datos y la carga de trabajo. Para lograr esta forma de distribución es fundamental abstraerse de la complejidad de la plataforma subyacente, lo cual se logra mediante el uso de una arquitectura de hardware distribuida donde implementar contenedores que interactúen entre si [19].

En esta forma de trabajar se necesita administrar y coordinar los contenedores (microservicios), para poder lograrlo se realiza la orquestación o coreografía de los mismos [20].

Orquestar significa que hay una entidad central (idealmente un microservicio en sí mismo) que coordina la comunicación entre los microservicios. Mientras que para microservicios coreografiados, cada servicio implicado en dicha coreografía "conoce" exactamente cuándo ejecutar sus operaciones y con quién debe interactuar, lo cual le quita transparencia e independencia a los servicios. Los servicios orquestados no "conocen" (y no necesitan conocer) que están implicados en un proceso de composición y que forman parte de un proceso de negocio de nivel más alto. Solamente el coordinador central de la orquestación es "consciente" de la meta a conseguir, por lo que la orquestación se centraliza mediante definiciones explícitas de las operaciones y del orden en el que se deben invocar los servicios.

Gracias a la orquestación de contenedores [21] se puede trabajar de forma transparente frente al aumento de carga de trabajo, falla de algún nodo, creación de nuevos contenedores, etc. El orquestador decide cuándo adaptar el sistema (por ejemplo, para iniciar la ejecución de un contenedor) y donde la adaptación tendrá lugar (por ejemplo, en qué nodo se ejecutarán los contenedores). Orquestar contenedores [22], aísla al usuario de los detalles de implementación, lo cual provee transparencia en el acceso y ejecución de los contenedores. Este aspecto es importante cuando se desea implementar replicación y distribución de datos masivos con el objetivo de lograr balanceo de carga y auto escalado horizontal, dependiente de la sobrecarga del volumen de información. Además, la orquestación, permitirá un aprovisionamiento de recursos bajo demanda y un escalado horizontal de la aplicación desarrollada.

Desarrollar aplicaciones de esta manera permitirá mitigar los problemas de continuidad de los datos provenientes del IoT para logar un procesamiento eficiente. Además, gracias a la portabilidad será posible alojar las aplicaciones en cualquier lugar desde la fuente de generación de los datos hasta el extremo (edge).

Líneas de Investigación, Desarrollo e Innovación

Las tareas de investigación se centrarán en la generación de buenas prácticas tendientes a lograr desarrollo de aplicaciones para big data, provenientes del IoT. Esto incluye el análisis de:

- Mejores prácticas de desarrollo basada en la filosofía DevOps.
- Métodos y herramientas de pre procesamiento y almacenamiento de grandes volúmenes de datos.
- Herramientas de contenerización de datos masivos.
- Definición de middleware de orquestación sobre arquitecturas distribuidas.
- Herramientas de medición de escalabilidad.

Resultados y Objetivos

Resultados Obtenidos

Durante los últimos dieciocho años se trabajó en una variedad de sistemas distribuidos y paralelos. Ya sea en Computación de Altas Prestaciones, en particular sobre análisis de diversas arquitecturas paralelas y distribuidas, tales como: Cloud Computing (públicos, híbridos y privados), Cluster de commodity, arquitecturas distribuidas de bajo costo y arquitecturas paralelas, como así también con arquitecturas de menores prestaciones (computación móvil y Edge Computing).

Dicha experiencia motivó la elaboración del proyecto de investigación del que forma parte la presente propuesta. El grupo ha realizado publicaciones en el área durante los últimos años: dieciocho trabajos de investigación en diferentes Congresos y Jornadas, se realizaron tres publicaciones en revistas científicas y se transfirieron los resultados mediante seis conferencias en eventos científicos y encuentros de divulgación.

Se han aprobado diecisiete tesinas de grado y un trabajo de especialización y se incorporaron dos becarios de investigación categoría alumno, un becario doctoral CONICET y se encuentran en desarrollo tesinas de grado y de maestría.

Objetivos

El objetivo del grupo de investigación es definir y construir una capa de orquestación de recursos, alineada con la filosofía DevOps, para brindar servicios de almacenamiento y procesamiento de grandes volúmenes de datos sobre arquitecturas distribuidas usando la tecnología de contenedores. Esta capa de orquestación deberá asegurar la continuidad mediante técnicas de replicación, distribución y balanceo de carga con el fin de maximizar la eficiencia de las aplicaciones.

Formación de Recursos Humanos

El equipo de trabajo está compuesto por los docentes-investigadores de la línea de investigación presentada que figuran en este trabajo. Cabe destacar que el proyecto marco de la presente línea de investigación incluye

investigadores de UNSL, Champagnat (Mendoza) y UNLaR.

Con respecto a la formación de recursos humanos, se está realizando una tesis doctoral, una tesis de maestría y cuatro tesinas de grado. Además este año se cuenta con un alumno el cual ha accedido a la beca CIN y cuyo tema se enmarca en la presente línea de investigación.

Por otro lado, se espera aumentar el número de publicaciones y también se prevé la divulgación de varios temas investigados por medio de cursos de postgrado y actualización o actividades de divulgación y asesoramiento a empresas y organismos del estado.

Referencias

[1] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," 2011.

[2] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," *Futur. Gener. Comput. Syst.*, vol. 29, no. 7, pp. 1645–1660, Sep. 2013.

[3] S. M. Babu, A. J. Lakshmi, and B. T. Rao, "A study on cloud based Internet of Things: CloudIoT," in *2015 Global Conference on Communication Technologies (GCCT)*, 2015, pp. 60–65.

[4] A. Cocchia, "Smart and Digital City: A Systematic Literature Review," Springer, Cham, 2014, pp. 13–43.

[5] N. R. Rodríguez *et al.*, "Orquestación de servicios para la continuidad edge al cloud," 2018.

[6] A. Katal, M. Wazid, and R. H. Goudar, "Big data: Issues, challenges, tools and Good practices," in *2013 Sixth International Conference on Contemporary Computing (IC3)*, 2013, pp. 404–409.

[7] M. Parashar, X. Li, and Wiley InterScience (Online service), *Advanced computational infrastructures for parallel and distributed adaptive applications*. John Wiley & Sons, 2010.

[8] M. A. Murazzo, M. J. Guevara, M. Tello, N. R. Rodríguez, F. Piccoli, and M. Giménez, "Orquestación de servicios para el desarrollo de aplicaciones para big data," 2018.

[9] M. Barrionuevo *et al.*, "Estrategias y análisis orientados al manejo de datos masivos usando computación de alto desempeño," 2018.

[10] I. Gorton and J. Klein, "Distribution, Data, Deployment: Software Architecture Convergence in Big Data Systems," *IEEE Softw.*, vol. 32, no. 3, pp. 78–85, May 2015.

[11] K. Bakshi, "Microservices-based software architecture and approaches," in *2017 IEEE Aerospace Conference*, 2017, pp. 1–8.

[12] N. Dragoni, I. Lanese, S. T. Larsen, M. Mazzara, R. Mustafin, and L. Safina, "Microservices: How To Make Your Application Scale," Springer, Cham, 2018, pp. 95–104.

[13] L. Zhu, L. Bass, and G. Champlin-Scharff, "DevOps and Its Practices," *IEEE Softw.*, vol. 33, no. 3, pp. 32–34, May 2016.

[14] N. Kratzke, "About Microservices, Containers and their Underestimated Impact on Network Performance," Sep. 2017.

[15] R. Morabito, J. Kjallman, and M. Komu, "Hypervisors vs. Lightweight Virtualization: A Performance Comparison," in *2015 IEEE International Conference on Cloud Engineering*, 2015, pp. 386–393.

[16] C. Ruiz, E. Jeanvoine, and L. Nussbaum, "Performance Evaluation of Containers for HPC," Springer, Cham, 2015, pp. 813–824.

[17] S. R. Brus, D. Wirasaet, J. J. Westerink, and C. Dawson, "Performance and Scalability Improvements for Discontinuous Galerkin Solutions to Conservation Laws on Unstructured Grids," *J. Sci. Comput.*, vol. 70, no. 1, pp. 210–242, Jan. 2017.

[18] S. Yangui, M. Mohamed, S. Tata, and S. Moalla, "Scalable Service Containers," in *2011 IEEE Third International Conference on Cloud Computing Technology and Science*, 2011, pp. 348–356.

[19] C. Pahl and B. Lee, "Containers and Clusters for Edge Cloud Architectures -- A Technology Review," in *2015 3rd International Conference on Future Internet of Things and Cloud*, 2015, pp. 379–386.

[20] N. Busi, R. Gorrieri, C. Guidi, R. Lucchi, and G. Zavattaro, "Choreography and Orchestration: A Synergic Approach for System Design," Springer, Berlin, Heidelberg, 2005, pp. 228–240.

[21] W. Gerlach, W. Tang, A. Wilke, D. Olson, and F. Meyer, "Container Orchestration for Scientific Workflows," in *2015 IEEE International Conference on Cloud Engineering*, 2015, pp. 377–378.

[22] J. Rufino, M. Alam, J. Ferreira, A. Rehman, and K. F. Tsang, "Orchestration of containerized microservices for IIoT using Docker," in *2017 IEEE International Conference on Industrial Technology (ICIT)*, 2017, pp. 1532–1536.